Audio Programming 2

Peter Dowsett

peter.dowsett@uwl.ac.uk

Today’s Lecture

- What are systems with Memory?
- What are Delay Line Effects?
- What is a circular buffers and why are they necessary?
    - Fractional delay
    - Interpolation
- Creating a delay effects in MATLAB

Systems with Memory

The gain and distortion algorithms we have written so far have all been *memoryless systems*.

This means that the system only depends on the input to the system at the same time. These systems do not require the memory of previous time samples to process a signal.

However, many signal processing systems used in audio where the output at any time can depend on the input signal at a previous time. These systems require memory to store the input samples for later use. These are *systems with memory*.

In fact, we have encountered a system with memory when we used parameter smoothing.

Delay Block Systems

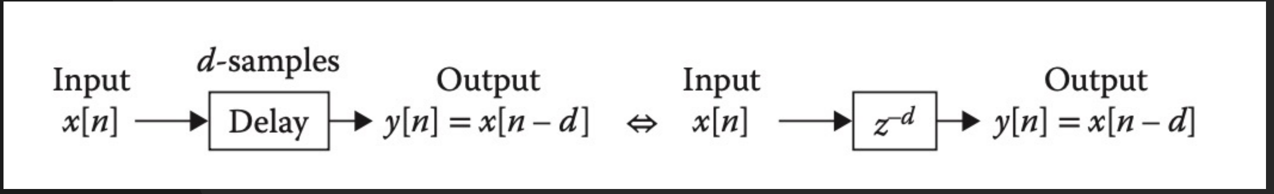
Systems with memory have the ability to temporarily store a signal and delay it over time.

A **delay** processing block receives a sample of the input signal at sample number *n* and stores this sample in memory for some number of samples, *d*.

At sample number *n* + *d*, the output of a delay processing block is the input sample, *x*[*n*].

Therefore, when input sample *x*[*n*] is received by a delay processing block, the output of the block at the same time is the sample *x*[*n* − *d*], or the sample stored in memory for *d* samples

In order to simplify the display of a block diagram, the notation z^{-d} is used to represent a delay block of *d* samples. This convention comes from the z-plane.



Delay Buffers

For a system to “remember” past audio samples you need to be able store them in a computers memory.

A collection of audio samples stored for the purpose of reading them back at a later date is called a ***delay buffer*** or ***delay line***.

Delay Line Effects

Systems that utilise delay blocks are often called *Delay Line Effects.* Delay line effects are a cornerstone of DSP and can be used creatively to make many effects that extend past audible delays, including:

Modulation effects - Phasing, flanging, Comb-Filtering, Chorus Spatial effects - Reverb. Spectral effects – EQ.

The creating of Modulation effects will be covered in Week 14. This week we will concentrate on audible delays.

Linear Buffer

One way to create a delay buffer is to add a new sample into the start of the buffer at every sample period and shift the rest of the samples one place towards the back of the buffer.

What are the problems with this approach?

Initial :	0	0	0	0	0	0	0
$n = 1$:	x_1	0	0	0	0	0	0
$n = 2$:	x_2	x_1	0	0	0	0	0
$n = 3$:	x_3	x_2	x_1	0	0	0	0
$n = 4$:	x_4	x_3	x_2	x_1	0	0	0
				⋮			
$n = 7$:	x_7	x_6	x_5	x_4	x_3	x_2	x_1
$n = 8$:	x_8	x_7	x_6	x_5	x_4	x_3	x_2

Exercise 1

Create a 5 sample-long delay using Linear Buffer in MATLAB.

This must use the the provided .wav impulse

Please graph the input and output signals to confirm your solution works.

Circular Buffers

One approach to avoid the inefficient task of shifting samples in a linear buffer is to use a **circular buffer**.

In a circular buffer, the value of each element is stored one time in its location and does not shift during an iteration.

Instead, the relative location of the start of the buffer is changed during an iteration. Therefore, it is only necessary to change one value (the buffer index) during an iteration instead of every element of the buffer.

The index of the delayed sample is determined relative to the current buffer index.

Circular Buffers Visually

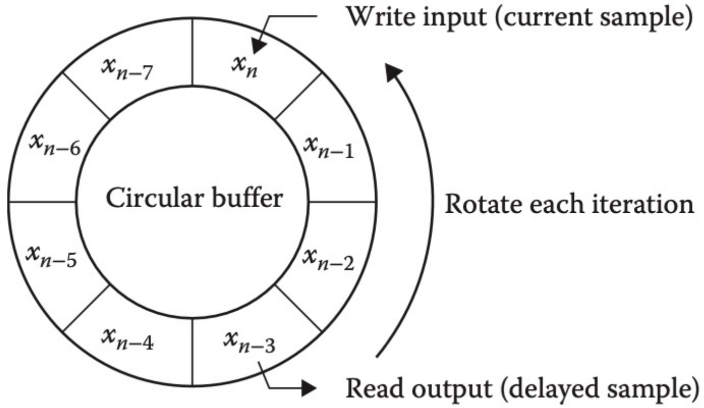
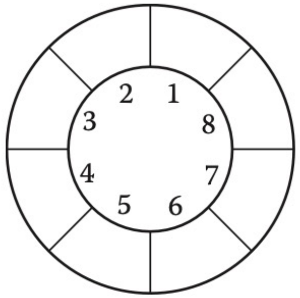
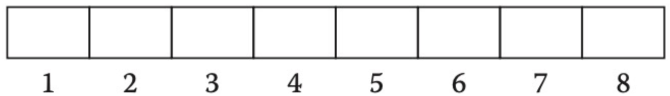


Figure 14.5: Indexing a circular delay buffer





Exercise 2

Create another 5 sample-long delay, using the provided .wav impulse, this time using a Circular Buffer in MATLAB.

Please graph the input and output signals to confirm your solution works.

Delay Line Maths

At a sampling rate of 48 kHz, how many samples do you need to delay a signal by if you want it to played back half a second later?

48,000 * 0.5 = 24,000 samples

How about 511.1 milliseconds (0.5111 seconds)?

48,000 * 0.5111 = 24532.8 samples

What is the problem?

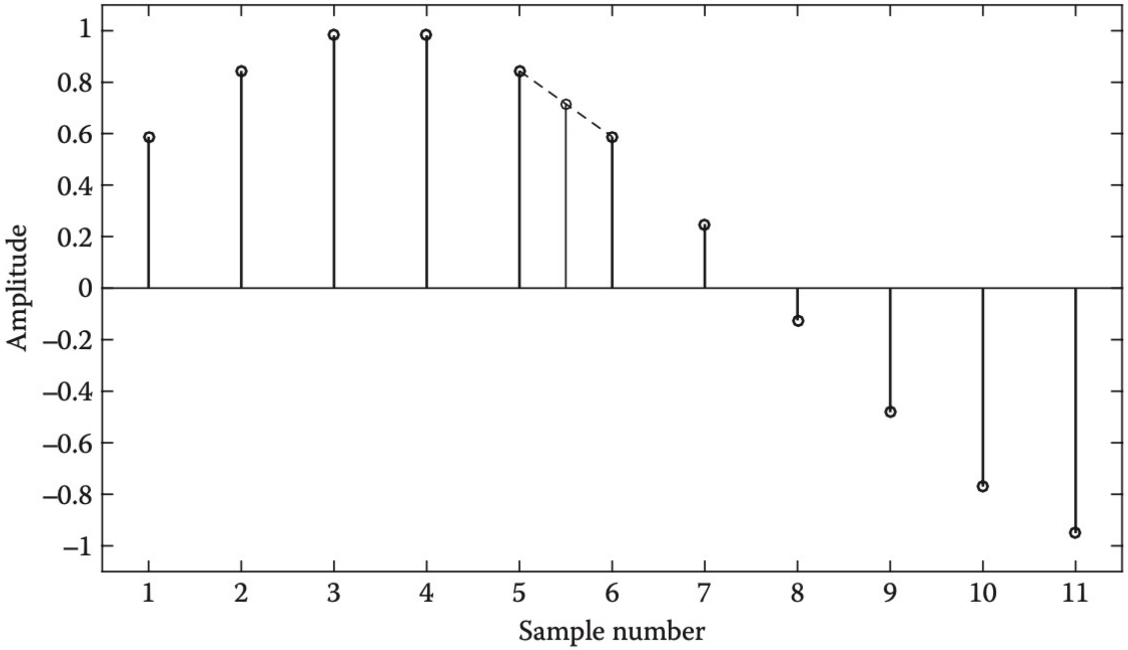
Fractional Delay

When a delay time falls between samples, you need a ***fractional delay line***.

To create a fractional delay line you need to use interpolation to get a value between samples.

There are many different types of interpolation algorithms, but the idea behind it are the same. Interpolation uses existing data to estimate the value of an unknown point.

In a delay line, this means using the samples surrounding the fractional point to estimate the value at that given moment in time.



$$(1-f_{rac})\cdot x_m+(f_{rac})\cdot x_{m+1}$$

Linear Interpolation

For linear interpolation, the value of a signal between samples is estimated by a linear combination of the two closest samples:

Cubic Interpolation

Cubic interpolation determines an estimated value of a signal between two samples by incorporating the amplitude of the four closest samples.

In theory, by incorporating additional samples in the estimate, then a more accurate estimate can be made. However, an increase in computational complexity is necessary, as each estimate requires additional processing steps

$$\text{Let: } a_0 = x_{m+2} - x_{m+1} - x_{m-1} + x_m$$

$$a_1 = x_{m-1} - x_m - a_0$$

$$a_2 = x_{m+1} - x_{m-1}$$

$$a_3 = x_m$$

$$y = a_0 \cdot (f_{rac}{}^3) + a_1 \cdot (f_{rac}{}^2) + a_2 \cdot f_{rac} + a_3$$

Exercise 3

Use the following tabulated data to calculate the value of a 5.67 sample delay for both linear and cubic interpolations.

Sample No	1	2	3	4	5	6	7	8	9
Value	0.2	0.432	0.12	0.432	0.2	0.786	0.98	0.777	0.45

$$(1-f_{rac})\cdot x_m + (f_{rac})\cdot x_{m+1}$$

Let: $a_0 = x_{m+2} - x_{m+1} - x_{m-1} + x_m$

$$a_1 = x_{m-1} - x_m - a_0$$

$$a_2 = x_{m+1} - x_{m-1}$$

$$a_3 = x_m$$

$$y = a_0 \cdot (f_{rac}{}^3) + a_1 \cdot (f_{rac}{}^2) + a_2 \cdot f_{rac} + a_3$$

Exercise 4

Create a MATLAB delay plug-in which implements a fractional delay line.

Additional Task: Make it switchable between linear and cubic interpolation.

Additional Task: Create a VST using the Audio Toolbox, as shown in the Prototyping lecture.